Low Bandwidth Zero Knowledge Authentication Protocol and Device

Field of the Invention

[01] The invention relates generally to authentication protocols, and more particularly to zero knowledge authentication protocols.

Background of the Invention

[02] Rustication

[03] Authentication is a process by which a prover assures a verifier that the prover has a claimed identity. A thorough description is provided by Menezes et al., "Handbook of applied cryptography," CRC Press, 1997, and Schneier "Applied cryptography," Wiley, 1996.

[04] Two primary objectives of the authentication protocol are completeness and soundness. Completeness means that the prover can *always* complete the authentication. Soundness means that an imposter has a *negligible* probability of completing the authentication.

[05] There are various grades of dishonesty and corresponding levels of security. In a conventional security framework, it is assumed that the imposter can make multiple attacks on the prover, Feige et al., "Zero knowledge proofs of identity," Journal of Cryptology, Vol. 1, pp. 77–94, 1988.

[06] A typical requirement of the authentication protocol is that the protocol is secure against impersonation under passive attack, where an adversarial prover has access to transcripts of prover-verifier interactions. A stronger requirement is that the protocol is secure against active attacks. Here, an adversarial prover can actively play the role of a cheating verifier with the prover numerous times before the impersonation attempt.

[07] It is also desired to secure against concurrent attacks. In these attacks, an adversarial prover plays the role of a cheating verifier prior to impersonation. A key distinction is that the imposter interacts concurrently with multiple honest provers.

[08] It is also desired that the authentication process does not reveal any secret information, e.g., the identity of the prover. This is known as zero knowledge authentication. A lucid and non-mathematical presentation of the concept of zero knowledge is provided by Quisquater et al., "*How to explain zero knowledge protocols to your children,*" Advances in Cryptology–CRYPTO '89, LNCS 435, pp. 628–631, 1989.

[09] With the zero knowledge authentication protocol, the prover convinces the verifier that the prover is in possession of knowledge of a secret, without revealing the secret itself. This is unlike a conventional password protocol, where the prover must reveal the secret to the verifier in order to authenticate.

[010] It is very important to keep in mind that the authentication protocol provides assurances only at the instant in time when the protocol is successfully completed. It is therefore important to ensure that the authentication process is tied

to some form of ongoing security service. At some level, all authentication protocols are vulnerable to the adversary who cuts in immediately after the successful authentication of the prover.

[011] Zero knowledge protocols allow the prover to demonstrate knowledge of the secret while revealing no information whatsoever other than the fact the prover possesses the secret, Goldwasser et al., "*The knowledge complexity of interactive proof-systems,*" Proceedings of the 17th Annual ACM Symposium on Theory of Computing, pp. 291–304, 1985, and U.S. Patent 4,995,082, "Method for identifying subscribers and for generating and verifying electronic signatures in a data exchange system," issued to Schnorr on February 19, 1991.

[012] An interactive protocol is said to be a proof of knowledge when there exists an algorithm that can always extract the secret. A protocol is said to be *zero knowledge* when it does not reveal the secret. As an advantage, the zero knowledge protocol does not suffer a degradation from repeated use and resists all attempts at impersonation.

[013] A protocol is said to be honest-verifier zero knowledge if it is zero knowledge when interacting with honest verifiers. The honest-verifier zero knowledge protocol has a weaker security guarantee than the general zero knowledge protocol because it is possible that a dishonest verifier can extract the secret.

[014] The ultimate measure of the worth of the authentication protocol lies in its security against impersonation attempts. A protocol that is secure against

impersonation under concurrent attacks is considered to be "secure" even if it is only honest-verifier zero knowledge.

[015] For a detailed and exhaustive discussions on authentication protocols including biometric techniques, see Davies et al., "*Security for computer networks,*" Wiley, 1989, and Ford, "*Computer communications security: principles, standard protocols and techniques,*" Prentice Hall, 1994. Password based schemes including the concept of salting are described by Morris et al., "*Password security: a case history,*" Communications of the ACM, Vol. 22, pp. 594–597, 1979, also see Needham et al., "*Using encryption for authentication in large networks of computers,*" Communications of the ACM, Vol. 21, pp. 993–999, 1978.

[016] Feige et al. adapted the concepts of zero knowledge to the application of authentication and proofs of knowledge. Since then there has been a veritable explosion of research into various aspects of zero knowledge, see Schneier "*Applied cryptography,*" Wiley, 1996.

[017] Conventional zero knowledge protocols require the interchange of a substantially amount of data between the prover and the verifier. The processing of the protocol also takes substantial computer resources. Therefore, zero proof knowledge protocols have only been feasible on high-complexity systems with a substantial communications bandwidth and memory, and a relatively sophisticated microprocessor. Such systems are relatively expensive and complex to build, operate and maintain.

[018] However, there are an increasing number of low-complexity, hand-held devices with severely limited resources. Many of these device use microcontrollers instead of microprocessors. For a microprocessor to be useful it must be attached to RAM, disks, keyboards, and other I/O peripherals. Operating systems and applications programs need to be installed. A microprocessor chip, by itself has no use.

[019] Not so for a microcontroller, which can be used as is. It usually has on-chip memory and for mass-produced controller, software is factory installed. As an advantage, microcontrollers are cheap to produce and simple to operate. There is no maintenance, because their low price makes them essentially disposable.

[020] Therefore, it is desired to provide a simple, low-bandwidth zero knowledge protocol, which can operate with microcontrollers.

Summary of the Invention

[021] The invention provides a generalization of Schnorr's zero knowledge authentication that is provably fast and secure. The invention permits the efficient batch verification of an ordered list of identities with various privilege classes. The invention use degree *d* polynomials to represent an the ordered list of *d* identities. The protocol is honest-verifier zero knowledge. In addition, the protocol is secure against impersonation under concurrent attacks.

[022] In a low-complexity authentication device, a microcontroller operates an LED to both transmit and detect light providing an optical communications channel for carrying authentication information.

[023] As an advantage, the device, unlike smart cards, does not require physical contact, so there is no wear. In addition, the device does not require an external power supply, subject to tampering, like a smart card.

[024] Unlike RF systems, the optical device is directional and short-range, so that the authentication environment is controlled. Also, a single device used as a key can open multiple locks without the fear of unintentionally opening a wrong lock that happens to be nearby.

[025] The visible nature of the LED allows for a natural user interface, i.e., the user can easily tell whether the LED is glowing or the battery is dead. In addition, the device can provide illumination, like a flashlight, while it is being used, providing additional physical security.

[026] Although the preferred device uses optical communications, the protocol can also be used with any other computing device, including smart cards, hand-held devices, and microprocessors.

Brief Description of the Drawings

[027] Figure 1 is a diagram of a zero proof authentication protocol according to the invention;

[028] Figure 2 is a schematic of a device that uses the protocol of Figure 1.

Detailed Description of the Preferred Embodiment

[029] The invention uses a generalization of Schnorr's zero knowledge authentication protocol, see the Appendix. A naïve generalization of Schnorr's scheme would authenticate d secret keys or *identities* by composing d authentication rounds in series. The prover sends d commitments, and the verifier replies d challenges, one for each identity. That scheme has a communication and computation cost that is d times the cost of Schnorr's scheme.

[030] In contrast, the authentication protocol according to the invention sends *one* commitment and the verifier replies *one* challenge for *all* identities, in parallel. The prover's reply is generalized from a degree one polynomial, according to Schnorr, to a degree d polynomial formed from d secret identities according to the invention.

[031] Figure 1 shows the basic steps of the authentication protocol according to the invention.

[032] The parameters used by the protocol according to the invention are as follows. Two prime numbers p and q are such that $q \mid p - 1$. A root of unity in Z_p of order q is a such that $\alpha \neq 1$, i.e., $\alpha^{q} \equiv 1 \ (mod\, p)$, or α is a generator of G_q. In other words, the values produced by $\alpha^q (mod\, p)$ eventually produces all integer values in the range 1 to G_q. The numbers p, q and α are publicly known.

[033] There are d_i secret identities 103. Each identity d_i consists of a private key s_i and a public key v_i. A private key s_i is a non-negative integer less than q,

chosen uniformly at random. A public key is $v_i = \alpha^{-si}$ $(mod\ p)$. Only the prover 101 knows the public keys.

[034] **Initiate**

[035] The prover 101 initiates 110 the protocol by sending an ordered list of the d public keys v, for which a prover claims to possess corresponding d private keys, to a verifier 102. It should be noted that the verifier can obtain the public keys 111 be other means. The net effect is that the verifier is in possession of the ordered d public keys.

[036] **Commitment by Prover**

[037] The prover selects a non-negative number r less than q uniformly at random. The prover sends 120 $x = \alpha^r$ $(mod\ p)$ 121 to the verifier.

[038] **Challenge from Verifier**

[039] The verifier selects a non-negative number e less than $2^{(t+log d)}$ uniformly at random. Here log is base 2. The number t is a security parameter, e.g. 95. The verifier sends 130 e 131 to the prover.

[040] **Response from Prover**

[041] The prover generates 104 $y = r + \Sigma_i\, s_i * e^i$ $(mod\ q)$, for $i = 1, \ldots, d$. The prover sends 140 y 141 to the verifier.

[042] Verification

[043] The verifier determines 105 if $x = \alpha^y * \Pi_i (v_i)^{e^i} (mod\, p)$ and accepts 150 the prover's claim if and only if the equality is true.

[044] Completeness

[045] It is clear that an honest prover can prove its identity to the verifier with a probability of one.

[046] Soundness and zero knowledge

[047] A fraudulent prover can cheat by guessing the correct challenge e ahead of time and sending the commitment $x = \alpha^r * \Pi_i (v_i)^{f^i} (mod\, p)$, where f is one of the at most d roots of the equation $r + \Sigma_i s_i * f^i = r + \Sigma_i s_i * e^i (mod\, q)$. Here, the response is chosen to be $y = r$. However the probability of success for this attack is at most 2^{-t}. This probability cannot be increased unless the discrete logarithm is easy to compute, which it is generally not the case.

[048] The corresponding theorem of Schnorr is extended, as described below, to show that the invented protocol is a proof of knowledge. If there exists a fraudulent prover with non-negligible probability of success, then by repeatedly simulating this fraudulent prover it is possible to uncover multiple successful transcripts with the same commitment. These transcripts can then be used to

compute all the *d* identities, thus demonstrating that the protocol is a proof of knowledge.

[049] The basic idea of the proof is that if $d + 1$ transcripts with the same commitment *r* are generated, then the discrete logarithm of each of the public keys can be determined by inverting the appropriate Van der Monde matrix, see Golub et al., "Matrix Computations," Johns Hopkins University Press, 1993.

[050] The description of the proof uses the following notation. The fraudulent prover *P'* is any probabilistic Turing machine that has the values *p*, *q* and α, as well as the *d* public keys v_i. A random string of *P'* is denoted by *RP*, an the success bit *S*(*RP*,*e*) is true (1) only if *P'* succeeds with *RP*, *e*, and false (0) otherwise.

[051] The success rate *S* is an average over *S*(*RP*,*e*), where *RP* and *e* are chosen uniformly at random. Let *T* be the running time of *P'*. It is assumed that *T* is independent of *RP* and *e* because limiting the time to twice the average running time for successful pairs *RP* and *e* decreases the success rate by at most a factor of 2.

[052] Theorem 1

[053] If the success rate *S* of *P'* is greater than 2^{-t+1}, then there exists a probabilistic Turing machine or process *PL* which, given black box access to *P'*, computes the *d* private keys $s_i = log_\alpha v_i$.

[054] Proof

[055] The process *PL* executes as follows.

[056] In step 1, the process selects the string *RP* at random and simulates *P'* using a random probe *e*, say *e1*. If *P'* fails then it repeats step 1 with a new *RP*. Otherwise it goes to step 2.

[057] In step 2, the process holds *RP* fixed and probes up to $(8u)(d+1)*log(d+1)$ random *e*'s in an attempt to find a total of $d + 1$ *e*'s, *e1*, *e2* ... $e(d+1)$ on which *P'* succeeds, where *u* is the number of passes of step 1. If the process fails in this attempt to find $d + 1$ *e*'s, then it goes back to step 1.

[058] If successful in this attempt, then the process solves the $d+1$ equations of the form $y = r + \Sigma_i\, si * e^i$ to compute the *d* unknowns *s1* through *sd*. The value *r* is the $(d+1)$st unknown. Solving these $d+1$ equations is equivalent to inverting the corresponding Van der Monde matrix.

[059] To analyze the running time of *PL*, we need some additional definitions and two auxiliary results. Define $S(RP)$ to be the fraction of e for which $S(RP,e)$ is *1*. Define *RP* to be "good" if $S(RP)$ is at least $S/2$. Let *#RP* denote the size of the set of all *RP* and *#e* the size of the set of all *e*. Note that $\#e = 2^{(t+logd)}$.

[060] **Lemma:** With probability at least ½, *PL* picks a good *RP* in step 1.

[061] **Proof:** The mean of $S(RP)$, over all *RP* chosen uniformly at random, is *S*. Now $\Sigma_{RP}\, S(RP) = \#RP * S$. But because $\Sigma_{not\text{-}good\, RP}\, S(RP) \leq \#RP * S/2$ it follows

that $\Sigma_{good\ RP}$ *S*(*RP*) $\geq$ #*RP* * *S*/2. In other words, the set of *RP*,*e*, for which *S*(*RP*,*e*) is *1 and RP* is good, is at least half the entire set for which *S*(*RP*,*e*) is *1*. Hence, *RP* is good with a probability of at least ½.

[062] **Lemma:**(Coupon collector lemma): With probability of at least ½, for a good *RP*, *PL* succeeds in finding a total of *d + 1 e*'s, *e1*, *e2* ... *e*(*d+1*) on which *P'* succeeds, using up to (*4/S*)(*d+1*)**log*(*d+1*) random probes.

[063] **Proof:** Fix the good RP. Observe that because RP is good there must be greater than S/2 e's such that S(RP, e) is 1, i.e. there must be greater than $2^{-t} * 2^{(t+\log d)} = d$ successful e's. Let there be $k \geq S/2 * 2^{(t+\log d)} \geq d + 1$ successful *e*'s, i.e., the *e*'s for which S(RP,e) is 1. Then, the expected number of probes to find *d* + 1 distinct successful e's is $2^{(t+\log d)}$ (1/*k* + 1/(*k*-1) + ... + 1/(*k* - *d*)). Because $k \geq S/2 * 2^{(t+\log d)}$ therefore the expected number of probes is at most (2*k*/*S*)(1/*k* + 1/(*k*-1) + ... + 1/(*k*)) which is at most (2/*S*)(*d*+1)ln(*d* +1). Hence with probability at least ½, PL will succeed using at most twice the expected number of probes.

[064] The expected number of probes in step 1 is 1/*S*, and the expectation of *u* is *1/S*, with probability at least ½, $u \geq (1/2)(1/S)$. *RP* is good with probability ½. Hence, with a probability of at least ¼, both $u \geq (1/2)(1/S)$ and *RP* are good, and *PL* succeeds in step 2 with a probability of at least ½. Because each probe takes *O*(*T*) steps, it follows that with probability at least *1/8*, *PL* succeeds in *O*(*dlogd* * *T/S*) steps.

[065] Hence, the expected time is bounded by ((*1/8*) + (*7/8*)(*1/8*) + (*7/8*)(*7/8*)(*1/8*) + ...) * *O*(*dlogd* * *T/S*) = *O*(*dlogd* * *T/S*) steps.

[066] In the case of a honest verifier, it is possible to generate transcripts with the same distribution. Hence, the protocol is honest-verifier zero knowledge. But it is not zero knowledge in the general case because a dishonest verifier could choose a challenge that is dependent on the commitment making it difficult to generate transcripts with the same distribution. Informally, however, the reason no information is revealed is that the numbers x and y, the commitment and the response, are essentially random.

[067] Security against impersonation under concurrent attack

[068] The invented protocol is secure against impersonation under concurrent attacks. As stated above, the ultimate end goal of the authentication protocol is establishing security against impersonation.

[069] In a concurrent attack, the adversarial prover is allowed to play the role of the cheating verifier and interact concurrently with multiple honest provers prior to the impersonation attempt.

[070] The proof is based on the assumption that discrete exponentiation is secure under d more inversions in the underlying group. Informally, the assumption states that if the cheating adversary is given access to a discrete logarithm oracle, then it is computationally infeasible for the adversary to compute the discrete logarithm of all challenge points in a randomly chosen collection with strictly d fewer queries than challenge points to the discrete logarithm oracle.

[071] If d is 1 and the collection of challenge points has size one, then this assumption is the usual one-wayness of the discrete exponentiation function. This assumption has the advantage of reducing the security of the authentication scheme to the hardness of a number theoretic problem. If $d > 1$, then the assumption is implied by, and hence no stronger than, a corresponding assumption described by Bellare et al., "GQ and Schnorr identification schemes: proofs of security against impersonation under active and concurrent attacks," Advances in Cryptology–CRYPTO '02, 2002.

[072] Before describing the proof of the theorem, the following notation is develop. Let *DLPG* denote the discrete logarithm parameter generator. This is a poly(k)-time randomized algorithm, given parameter k outputs the pair q and α, where q is prime such that $q \mid p - 1$, and α is a generator of G_q; p is k bits long. The proof does not require a specific generator. The proof of the security of the protocol is based on an assumption about *DLPG*.

[073] A *dmdl* adversary is a randomized polynomial-time algorithm that gets input q and α and has access to two oracles – a challenge oracle that returns a random element in G_q each time it is invoked, and a discrete logarithm oracle that, given an element Y in G_q, returns $log_{\alpha}Y$. The game is to run *DLPG* and then to run the *dmdl* adversary on its output. The *omdl* adversary wins if it successfully finds the discrete logarithms of all the points generated by the challenge oracle while querying the discrete logarithm oracle on strictly d fewer points.

[074] Let *adv-dmdl*(k) denote the success probability of the *dmdl* adversary. Let *impca* denote the adversary that first takes on the role of fraudulent verifier and interacts concurrently with several honest prover clones before subsequently taking on the role of fraudulent prover. Let *adv-impca*(k) denote the probability that *impca* is successful at impersonation.

[075] **Theorem 2**

[076] Given an adversary *impca* attacking the protocol associated with *DLPG*, there exists an *dmdl* adversary such that for every k
$Adv\text{-}impca(k) <= 2^{-t} + (adv\text{-}dmdl(k))^{1/(d+1)}$

[077] Proof

[078] It is possible to construct a *dmdl* adversary that interacts with the *impca* adversary and uses the knowledge obtained to compute the discrete logarithm of d more challenge points than it obtains from the discrete logarithm oracle.

[079] First, the *dmdl* adversary queries the challenge oracle d times and obtains d random group elements $v_i = \alpha^{-si}$. Then, the *dmdl* adversary runs the *impca* adversary in cheating verifier mode and corresponding to each prover returns commitments x_i by querying the challenge oracle and responses y_i, corresponding to challenges e_i, by returning the response from the discrete logarithm oracle to the query $x_i * \Pi_j (v_i)(-e_i^j), j = 1$ to d.

[080] Second, the *dmdl* adversary runs the *impca* adversary in cheating prover mode $d + 1$ times. If any two challenges are the same, then the *dmdl* adversary

fails. The commitment of the cheating prover stays the same, because it starts each time with the knowledge gained as cheating verifier. Any randomness can also be incorporated into this knowledge.

[081] Let $x = \alpha^r$ be the commitment and let w_i be the response corresponding to the distinct challenges c_i. If the cheating prover fails even once then, the *dmdl* adversary fails. If the cheating prover succeeds each of the $d + 1$ times, then the *dmdl* adversary has $d + 1$ equations of the form $w_i = r + \Sigma_j s_j * c_i^j$, with $d + 1$ unknowns, r and the d c_i. By inverting the Van der Monde matrix formed from these equations, the equations can be solved to obtain the s_i. Then, the *dmdl* adversary returns $y_i - \Sigma_j s_j * e_i^j$ corresponding to the query x_i made of the challenge oracle.

[082] It is possible to verify that *dmdl* succeeds with strictly *d* fewer queries to the discrete logarithm oracle when the cheating *impca* prover succeeds $d + 1$ times. The probabilities of the two events are now related. An auxiliary result is a generalization of an equivalent result obtained by Bellare et al.

[083] **Generalized Reset Lemma:** Consider any prover, including potentially cheating provers. Let *A* and *B* be random variables over the space of the random coins, RP, of the prover. Let A denote the probability, taken over e, that the verifier accepts. Let B denote the probability, taken over e's, that when the verifier is reset and run d + 1 times, a different e is generated each time and the verifier accepts each time. Let acc = E(A) and res = E(B). Then $\text{acc} \leq 2^{-t} + \text{res}^{1/d+1}$.

[084] **Proof:** Let $1/c = 2^{t + \log d}$ be the size of the challenge set, i.e. #e. It is easy to see that $B \geq A(A-c)(A-2c) \ldots (A-dc)$. This implies that $B \geq (A-dc)^{(d+1)}$ which yields that $E(A) \leq dc + E(B)^{1/(d+1)}$ or $E(A) \leq 2^{-t} + E(B)^{1/d+1}$.

[085] Now observe that *adv-impca*(k) = E(*acc*), where the expectation is taken over the choice of v_i and the knowledge gained as the cheating verifier. Similarly,

$\leq$ *adv-dmdl*(k) = E(*res*).

Applying the reset lemma,

adv-impca(k) = E(*acc*)

$\leq$ $E(2^{-t} + (res)^{1/(d+1)})$

1. $= 2^{-t} + E((res)^{1/(d+1)})$

$\leq$ $2^{-t} + (E(res))^{1/(d+1)}$, (by Jensen's inequality), and

1. $= 2^{-t} + (adv\text{-}dmdl)^{1/(d+1)}$

[086] The following corollary is a straightforward consequence of Theorem 2.

[087] Corollary

[088] If *DLPG* is such that *adv-dmdl* is negligible and if $t = \omega(\log k)$, then the protocol associated with *DLPG* is secure against concurrent attacks. The assumption that t is super-logarithmic in k is necessary for otherwise the scheme can be broken by guessing the verifier's challenge.

[089] Authentication using Multiple Identities

[090] Authentication protocols form the building blocks of authorization and access control systems. A typical structure of an access control system includes an access control matrix of resources and associated identities. A successful authentication of an identity enables access to the associated resources.

[091] There are two primary types of access control. In one type, the prover has a certificate that authorizes the prover access to a particular set of resources. In the other type, the verifier possesses information regarding allowed resources for a particular identity.

[092] In practical applications, provers do not always possess the wherewithal to engage in certificate exchange, and verifiers are often stand-alone and hence need to be provided with a list of allowed identities and corresponding privileges.

[093] In theory, one could generate a unique identity for each subset of resource and provide verifiers with the access control matrix corresponding to all possible identities. However, this is prohibitively expensive in terms of storage because there are 2^n possible resource subsets for n resources.

[094] However, it is possible to take advantage of the fact that the present protocol can authenticate an ordered list of up to d identities in one round. Then, given a memory at the verifier that can support up to m identities, i.e., public/private key pairs, one can generate $1 + m + m^2 + \ldots m^d = (m^{d+1} - 1)/(m - 1)$ possible resource sets.

[095] Thus, even if there is insufficient space for 2^n distinct identities, it is possible to achieve the same effect by selecting an appropriate $d = O(n/\log m)$, by trading off processing and memory.

[096] Efficiency

[097] For an ordered list of d identities, the authentication protocol according to the invention $O(\log d)$ has more bits than Schnorr's scheme for one identity. However, it requires d modular multiplications, whereas Schnorr's scheme makes do with only one modular multiplication.

[098] Implementations

[099] LEDs

[0100] Light emitting diodes (LEDs) are ubiquitous interfaces. Their diverse applications include numeric displays, flashlights, vehicle brake lights, traffic signals and the omni-present equipment indicator.

[0101] Because LEDs are commonly used as light emitters, it is often forget that LEDs are fundamentally photodiodes, and hence can also operate as light detectors. Although LEDs are not optimized for light detection, LEDs can be very effective at detecting light.

[0102] The interchangeability between solid-state light emission and detection was described by Forrest W. Mims in "Siliconnections: Coming of age in the electronic era," McGraw Hill, 1986. and "LED circuits and projects," Sams and Co, 1993, but has since largely been forgotten.

[0103] An interface circuit for alternately emitting and detecting light using an LED is described by Dietz et al., in U.S. Patent Application 10/126,761 filed by Dietz et al. on April 19, 2001, In addition to the LED and two digital I/O pins of a simple microcontroller, the circuit requires only a single current limiting resistor. When forward-biased, the LED emits light, and when reverse-biased, the LED it detects light.

[0104] The implications of LED-based data communication are significant, because it is essentially software that makes the microcontroller operate the LED in two different modes. With this technique any LED connected to a microprocessor can be made to operate as a bi-directional data port, although at a relatively low bandwidth. The micro-processor is similarly constrained in its processing capabilities.

[0105] Hardware

[0106] The low-bandwidth zero knowledge authentication protocol can be implemented with a relatively simple device 200 as shown in Figure 2.

[0107] The authentication device 200 includes a microcontroller 210, e.g., a Microchip PIC16LF628. A first I/O pin 201 of the microcontroller is connected to an input of an LED 220, and a second I/O pin 202 is connected to a resister 230 connected, in turn, to an output of the LED. A push button switch 240 connected to a third I/O pin 203 operates the device from a small a 3-volt lithium coin-cell battery 250. All other components shown operate conventionally. During operation, the device emits and senses light 250 as described by Dietz et al. to transmit and receive data corresponding to the authentication protocol described herein.

[0108] The PIC uses 8-bit instructions, runs at 5 MIPS, and has 16KB of writeable memory. A mass produced version of the device 200 would cost no more than a LED keychain flashlight. In fact, the device according to the invention can also be used a flashlight.

[0109] Because the data rate of the device is relatively low, the low bandwidth authentication protocol is ideally suited for the device. The range of communication is several centimeters. The intent is that the prover operates the device. The verifier uses a similar device with, perhaps, a more powerful microcontroller or microprocessor.

[0110] In a typical application, the prover is a person that desires to gain access to a secure facility (house, bank) or equipment (ATM, car, computer) controlled by the verifier.

[0111] Security

[0112] The security parameter t, see above, is set to *95*. This is generally considered adequate security, see Schnorr. The number of identities d is set to *32* so that $t + \log d = 100$.

[0113] The prime number q has at least *200* bits long because of the existence of an $O(q^{1/2})$ 'baby-step/giant-step' process for finding discrete logarithms, see Schnorr. In conjunction with the existence of a general number field sieve, the prime number p has about 1500 bits, see LaMacchia et al., "*Computation of discrete logarithms in prime fields,*" Designs, Codes and Cryptography, Vol. 1, pp. 46–62, 1991.

[0114] Prover

[0115] As an advantage, the protocol according to the invention can operate with a relatively small amount of memory. The small sized memory stores both the data, e.g., keys, parameter, and tables, and the software implemented protocol. This is common for small sized devices, as well as smart cards, which can also use the invention, With smart cards, the data are communicated electrically, instead of optically, requiring a physical interface.

[0116] The primary operation performed by the prover is modular multiplication. This multiplication can be implemented by the well known fast Fourier transform of Cooley and Tukey, which takes $O(nlogn)$ bit operations, see Strang G., "Linear algebra and its applications," Harcourt Brace, 1988. However, that scheme has a relatively high complexity, even though it is asymptotically efficient.

[0117] Hence, the preferred embodiment uses a pre-computed table to substantially reduce the complexity of the procedure as well as computation time. For each private key v, the memory stores a pre-computed table of the residues modulo q of the product of the private key with the powers of 2 from 2^{l} to $2^{l + \log q}$.

[0118] To multiply the private key with any given number, the residues corresponding to the powers of 2 present in the binary representation of that number are added. The residue modulo q of $2^{l + \log q}$ enables one to reduce the overflow when doing addition, so that one always has a number with log q bits.

[0119] In response to receiving the challenge e 131 from the verifier, a similar table storing the residues modulo q of the product of e with the powers of 2 from 2^l to $2^{l + \log q}$ is constructed. This table is used to determine the powers of e, and the pre-computed tables of the private keys is used to compute $y = r + \Sigma_i s_i * e^i \ (mod\ q)$. This enhancement enables the device to complete the authentication in a matter of seconds or shorter depending on size of the various parameters.

[0120] Smart-Cards

[0121] The invention can also be used with wallet-sized smart cards. Smart-cards are usable for any number of applications. In the past, they are typically used to spend and receive electronic cash and credit. They can also be used to make telephone calls from payphones, sign documents, access bank accounts or gain entry to buildings. They are typically tamper-proof, see Guillou et al., "The smart card: a standardized security device dedicated to public cryptography," Contemporary Cryptology: The Science of Information Integrity, IEEE, pp. 561–613, 1992.

[0122] Smart cards typically consist of a single microcontroller that includes a memory (RAM and PROM) to store the necessary parameters for the execution of public key schemes. Many smart cards systems employ magnetic stripes and card readers for communication. Smart cards typically rely on external power sources, e.g. from the card reader. This makes it harder to render smart cards tamper-proof against hostile environments with compromised power supplies. These microcontroller can be programmed to implement to protocol according to the invention.

Effect of the Invention

[0123] The invention provides a low-complexity zero knowledge authentication protocol that can be used with low-complexity portable device, such as hand-held optical transceivers, and smart cards. The protocol is secure against impersonation under active and concurrent attacks. The invented protocol is honest-verifier zero knowledge, and can be extended to a general zero knowledge protocol.

[0124] Although the invention has been described by way of examples of preferred embodiments, it is to be understood that various other adaptations and modifications can be made within the spirit and scope of the invention. Therefore, it is the object of the appended claims to cover all such variations and modifications as come within the true spirit and scope of the invention.